Formal Design of Performance-driven Self-adaptive Systems under Uncertainty

Emilio Incerto, Mirco Tribastone, Catia Trubiani

IMT School for Advanced Studies, Lucca

InfQ2018
22th November 2018, Milan

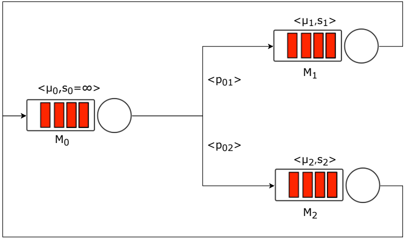
Motivation

- Performance characterization it is not an easy task
  - Deep knowledge about the system and its deploying platform
  - Affected by different kind of uncertainty
  - Non linear behaviour
- Try now fix later, what if analysis
- They fail at runtime due to the strong variability of the execution conditions

Performance-driven self-adaptation is promising

- Monitors the system execution
- Continuously updates a model of the system under study
- Triggers reconfiguration when required

The Queuing Networks Fluid Approximation

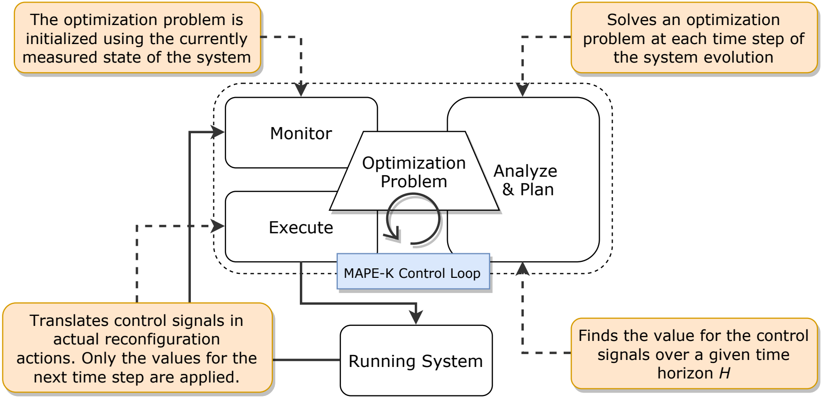


Queueing network model for a load balancing system

$$\Downarrow$$

$$\dot{x}_0(t) = -\mu_0 \min\{x_0(t), s_0\} + \mu_1 \min\{x_1(t), s_1\} + \mu_2 \min\{x_2(t), s_2\}$$
$$\dot{x}_1(t) = -\mu_1 \min\{x_1(t), s_1\} + p_{0,1}\mu_0 \min\{x_0(t), s_0\}$$
$$\dot{x}_2(t) = -\mu_2 \min\{x_2(t), s_2\} + p_{0,2}\mu_0 \min\{x_0(t), s_0\}$$

Performance-driven Adaptation Philosophy



The main idea is to encode the discrete time version the ODE model as constraints of the optimization problem

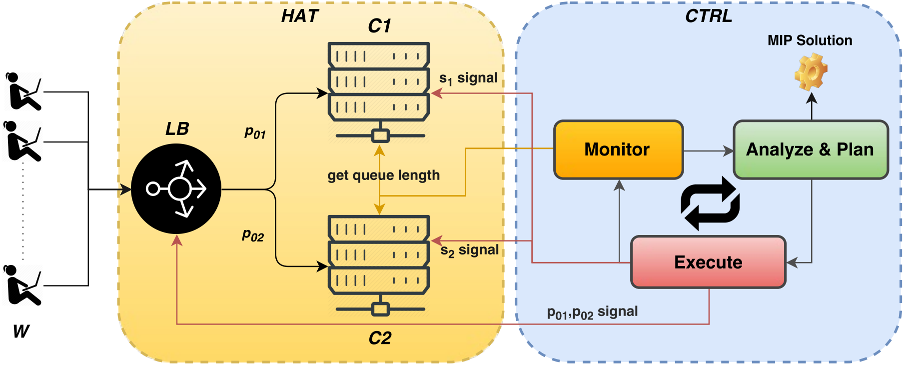
Efficient MPC Performance Adaptation[1]

- **Focus**: performance driven self-adaptation for CPU bound applications
- Fluid queuing networks as the enabling technology
- Model predictive control (MPC) for exploring the adaptation space:
  - fully automated
  - multiple adaptation knobs
  - considers actual run-time conditions
  - involves the solution of Mixed Integer Non Linear Programs (MINLPs)

As a main technical result we formally translate the naive MINLP MPC formulation in a Mixed Integer Programming (MIP) one

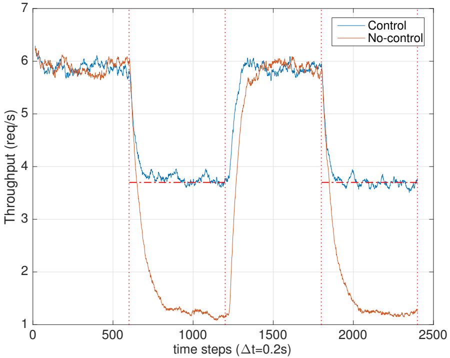
[1] Software Performance Self-adaptation through Efficient Model Predictive Control, Emilio Incerto, Mirco Tribastone and Catia Trubiani (ASE 2017)

Numerical Evaluation: HAT architecture

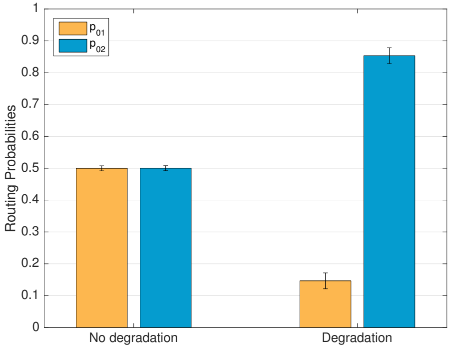


HAT architecture

Numerical Evaluation: Hardware degradation



(a) System throughput



(b) Optimal control signals

Hardware degradation experiment

Numerical Evaluation: Scalability

Comparison with markov decision processes (TO: timeout after 120 s)

	MIP	*Markov Decision Processes*		
W	Runtime(s)	Runtime(s)	# States	# Transitions
80	0.0037	71	3 018 789	334 732 743
90	0.0036	87	3 805 074	421 958 628
100	0.0040	TO	4 682 259	519 272 613
110	0.0038	TO	5 650 344	626 674 698
120	0.0041	TO	6 709 329	744 164 883

Estimation of Service Demands in Queuing Networks[2]:Introduction

- Well calibrated model parameters are necessary for computing accurate predictions
- When dealing with queuing networks service demands are fundamental
- The estimation need to be performed:
  - continuously
  - non intrusively

As a main technical result we formulate the estimation problem as a Quadratic Programming (QP) one solved according to a Moving Horizon paradigm

[2] Moving Horizon Estimation of Service Demands in Queuing Networks, Emilio Incerto, Annalisa Napolitano and Mirco Tribastone (MASCOTS 2018)

Estimation of Service Demands in Queuing Networks: Evaluation

Accuracy comparison between the queue length maximum likelihood estimation (QMLE) and our approach (MHE).

	$x(0) = (3, 0, 0)$ $H = 2347, U_2 \approx 0.10$		$x(0) = (9, 0, 0)$ $H = 688, U_2 \approx 0.30$		$x(0) = (12, 0, 0)$ $H = 521, U_2 \approx 0.40$		$x(0) = (19, 0, 0)$ $H = 353, U_2 \approx 0.60$		$x(0) = (26, 0, 0)$ $H = 262, U_2 \approx 0.80$	
K	QMLE	MHE	QMLE	MHE	QMLE	MHE	QMLE	MHE	QMLE	MHE
1	0.52	9.25 ± 1.03	1.37	9.63 ± 1.06	2.07	7.90 ± 1.01	3.40	6.58 ± 0.81	5.15	4.89 ± 0.69
2	448.30	4.13 ± 0.62	126.54	3.93 ± 0.58	67.18	4.20 ± 0.63	5.46	3.90 ± 0.56	2.33	3.59 ± 0.54
5	184.02	2.26 ± 0.33	60.41	3.02 ± 0.43	42.09	2.76 ± 0.38	8.78	2.07 ± 0.33	1.65	2.06 ± 0.34
10	92.29	1.65 ± 0.27	30.53	1.99 ± 0.31	23.18	1.82 ± 0.31	9.50	2.09 ± 0.30	3.89	1.50 ± 0.24
20	45.18	1.37 ± 0.21	15.01	1.13 ± 0.19	11.32	1.36 ± 0.18	6.41	1.36 ± 0.19	5.81	1.17 ± 0.18
50	18.67	0.74 ± 0.10	6.08	0.81 ± 0.14	4.57	0.78 ± 0.11	2.72	0.81 ± 0.12	5.17	0.73 ± 0.10

Future Works

- More expressive MPC control formulation based on the fluid interpretation of layered queuing networks (LQNs)
- Higher order performance-driven adaptation techniques.
- Estimating generally distributed service demands
- Learning LQNs model
- Shifting the focus on quantitative properties of the code

Future Works

- More expressive MPC control formulation based on the fluid interpretation of layered queuing networks (LQNs)
- Higher order performance-driven adaptation techniques.
- Estimating generally distributed service demands
- Learning LQNs model
- Shifting the focus on quantitative properties of the code

Questions?